ROBUST GENERATION OF QUADRILATERAL/PRISMATIC BOUNDARY LAYER MESHES BASED ON RIGID MAPPING

Hongfei Ye[1] Taoran Liu[2] Jianjun Chen*[3] Yao Zheng[3]

[1] *Centor for Engineering and Scientific Computation, Zhejiang University, Hangzhou310027, Zhejiang, China. hfye@zju.edu.cn*
[2] *Centor for Engineering and Scientific Computation, Zhejiang University, Hangzhou310027, Zhejiang, China. taoranliu@zju.edu.cn*
[3] *Centor for Engineering and Scientific Computation, Zhejiang University, Hangzhou310027, Zhejiang, China. chenjj@zju.edu.cn*
[4] *Centor for Engineering and Scientific Computation, Zhejiang University, Hangzhou310027, Zhejiang, China. yao.zheng@zju.edu.cn*

ABSTRACT

Quadrilateral/prismatic boundary layer meshes are believed to combine precision with ease of use. However, the generation of full-layer boundary layer meshes without transition elements still needs to encounter the problem of robustness. This paper proposes a novel and robust full-layer boundary layer mesh generation scheme, which constructs an orthogonal target mesh and an extremely thin initial mesh and then iterates the initial mesh until its rigid mapping energy to the target mesh is minimal. A positive area/volume-preserving rigid mapping method is applied iteratively to ensure robustness. This method has been partly validated in 2D and has achieved preliminary results in 3D.

Keywords: anisotropy mesh, mesh generation, rigid mapping, boundary layer mesh

1. INTRODUCTION

1.1 Prismatic Mesh Generation

A boundary layer mesh is a semi-structured layered mesh around a given geometry. The early generation methods of boundary layer meshes are mainly PDE-Based methods, which have been widely used in early structured mesh methods [1, 2, 3, 4]. Later, as the geometry model became more and more complex, the semi-structured mesh gradually developed, and a separate boundary layer mesh concept was gradually formed, along with the unstructured mesh filled in between the boundary layer mesh and the bounding box.

Among different schemes of meshes for solving partial differential equations (PDE) by numerical methods near the boundary, the generation of a layered prismatic(in 3D) and quadrilateral(in 2D) mesh with isotropic mesh has gained popularity due to its good compromise between viscous accuracy and ease of use [5]. In this mesh, layered elements are configured on the near field of viscous walls to resolve high flow gradients normal to the walls. In contrast, the remaining domain and the surface geometry are filled with unstructured meshes.

The most widely applicable method for generating layered meshes is the *Advancing Layer Method* (ALM). This method is usually generated in a layered manner, and premature stopping caused by global intersections may occur, requiring pyramid transition elements to handle mesh continuity. Generally, pyramids used for transitions are highly twisted, and their exposed faces are not conducive to the following isotropic mesh generation. Therefore, full-layer boundary layer mesh without transition elements is more sought after. However, full-layer generation usually encounters problems, the most serious of which is the negative volume cell. This paper proposes a global method with strictly positive volume guarantees for generating full-layer boundary layer mesh under arbitrary input. One of the resulting meshes in 2D is shown in Figure 1. We can see that the algorithm handles the narrow gap well and do well in both the boundary layer mesh completeness and normal orthogonality. Here mesh completeness indicates the area/volume covered by the boundary layer mesh. Usually, the larger the boundary layer region is covered, the higher the accuracy of the solution.

The global technique entails solving marching normal information globally, typically via the use of a set of linear equations or numerical methods. Some of the study [6, 7] still rely on or partly rely on the ALM framework, and some [8, 9] do not. The widely-recognized advantage of the global method is that its normals are globally optimized. Practically, the shortcomings of this method are also pronounced: 1. the technique is usually time-consuming, whether for the explicit or implicit way. 2. since the normal is globally optimized, unsuitable normal may be generated locally, such as the singularity [8] or negative elements.

PDE-based One of the methods for normal smoothing is the PDE-governed approach, and the equation is often solved by the implicit method. The PDE-based method provides a new global angle to view the normal smoothing problem, such as Laplacian equation [7], Eikonal equation [9, 10] and level set equation [11], which models ALM as a hyperbolic differential equation. The computation of marching normal directions is defined in the solution space of the adopted governing equation. Since the solution is smooth in the flow domain, the marching normal is naturally smoothed following the corresponding equation. For instance, the marching direction at a point could be defined as the gradient vector of the solution proposed by Wang *et al.* [12], based on a variation of Eikonal equation solution about minimum Euclidean distance. Zheng *et al.* [7] proposed a method to solve the Laplace equation of three components of marching normal respectively by *boundary element method* (BEM).

Variation-based Another global method is optimizing regeneration elements globally based on the variational method, which relies on a valid initial mesh. Variation-based methods are usually solved by explicit methods. This approach relies on a partial [6]or full [13] background mesh and then achieves both mesh orthogonality and mesh untangling in a quality-optimized manner. In general, weighted energy will be defined, including orthogonality and normal smoothing energy, and minimizing the energy will be used to obtain the normals. Two typical applications of this approach are the method proposed by Dyedov *et al.* [13], which minimizes the control triangle shapes energy and side-edge orthogonality. Garanzha *et al.* [6], which minimizes the objective function relates to the Jacobian matrix of all prisms from Lagrangian coordinates to Eulerian coordinates.

Local Method The local method means there is no global function solved during the normal calculation, and smoothing is performed on each layer or each cell. The smoothing is often locally optimized by authors' extensive experience [14, 15, 16, 17]. The biggest advantage of the local method is its high flexibility in the normal direction so that it perfectly coincides with the idea of local greed in the ALM. Moreover, the process is usually not time-consuming since the smoothing is performed locally. However, since the smoothing problem is usually non-convex, the local optimal usually cannot lead to the global optimal. Herefore, the final mesh may need more advantages of the global method, such as mesh completeness. Loseille *et al.* [18] proposed a 3D local operator that combines several local topology operations and uses it to generate an anisotropic boundary layer tetrahedron mesh.

1.2 Rigid Transformation

Rigid Transformation(also called Euclidean transformation or Euclidean isometry) is a geometric transformation of a Euclidean space that preserves the metrics of Euler spaces [19]. This concept is widely studied in computer graphics, especially parameterization [20, 21, 22] and mesh deformation [23], and also shape interpolation [24, 25]. Similar to the application of boundary layer meshes, the most difficult goal in the study of rigid transformations is flip-free mapping with non-intersecting boundaries, also known as *bijective* in the field of surface parameterization. However, boundary layer mesh generation is more complex than parameterization because the quality of the initial mesh usually needs improvement. In addition, the application of air mesh can handle the self-intersection of rigid mapping at the free outer boundary [21]. The idea of air mesh is straightforward. The air mesh is an isotropic tetrahedral mesh between the outermost triangle mesh of the boundary layer mesh and the bounding box. When the mesh is deformed, the fold-free isotropic mesh is equivalent to the self-intersection-free boundary layer mesh. After that, Müller *et al.* [26]extends the technique of [27] to add the concept of triangle flipping based on a quality measure during the optimization instead of retriangulating the air mesh.

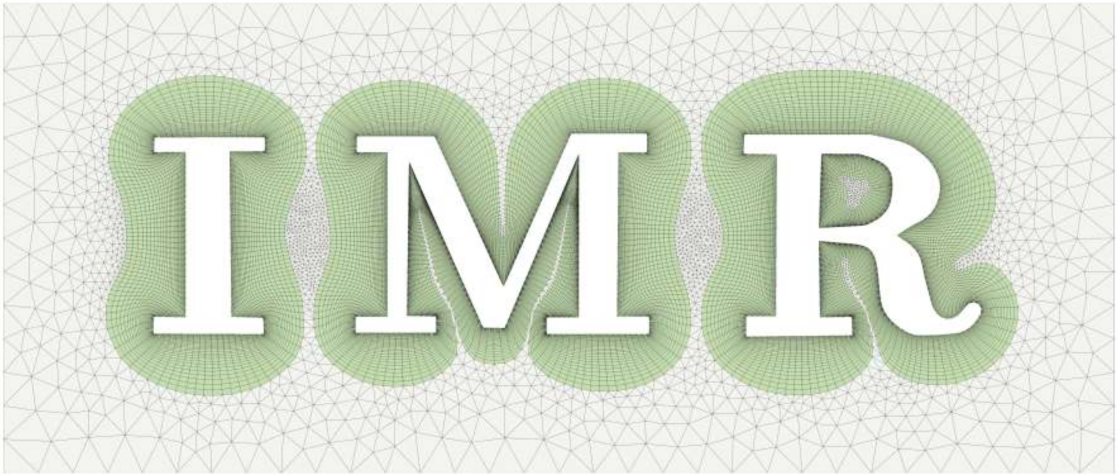


Figure 1: The final 2D viscous mesh of three-letter model generated by the proposed method. Full-Layered boundary layer meshes are colored green.

1.3 Contribution

This paper proposes a robust boundary layer mesh generation algorithm based on rigid mapping that has been partly validated in 2D and 3D. This work is mainly inspired by surface parameterization. Our contribution can be listed below:

1. Innovatively introduces the rigid mapping into layered boundary layer mesh generation, along with the air mesh technique, which is used to prevent negative elements. The introduction of these technologies makes high-quality full-layer boundary layer mesh generation with positive volumes guaranteed under arbitrary input theoretically possible.

2. This paper proposes the generation scheme of the target mesh and the initial mesh of rigid transformation. Besides, by introducing an adaptive vertical target mesh adjustment and multiple normals configuration, the quality of the boundary layer mesh has been significantly improved.

3. The experimental 2D version of the algorithm open sourced at github[1].It is encouraging that the 3D version of the algorithm has achieved good preliminary test results.

2. METHODS OVERVIEW

Figure 2 presents the proposed workflow of 2D layered boundary layer mesh generation. It inputs a *Planar Straight line Graph* (PSLG) and a few user parameters defining the preferred property of the output mesh. A typical set of these user parameters includes the height of the first layer and the ratio between the heights of neighboring layers. In addition, the loops in PSLG should be properly wound out by the *winding number* [28] to determine the direction of boundary layer growth.

1. **Initial Mesh** Iteration start from the initial mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$, where $\mathcal{V}$ is the set of vertices' coordinate and $\mathcal{F}$ is the set of all connection between vertexes. First, marching normals are defined on each node in PSLG as the average front normal of both edges that share this node, which ensures normal visibility. Second, an extremely small initial marching step length is defined, which must be small enough to free the mesh from global and local intersections. The value can be given by the user and obtained by the dichotomy method. Third, a layered quadrilateral mesh with the same number of layers is generated with respect to the user input parameters and the marching normal.

2. **Target Mesh** The target mesh $\mathcal{M}' = (\mathcal{V}', \mathcal{F}')$ defines our "expectation" for boundary layer mesh. For each segment in PSLG, two normals orthogonal to the segment are generated, and the length of the two normals are identical, which is specified by the user. Then, two normals and the segment can form a rectangle. Layer by layer, the entire target mesh is generated with respect to the user input parameters. As illustrated in Figure 2, the target mesh cells are all rectangular

[1] https://github.com/HongviYe/2D-viscous-mesh-generation

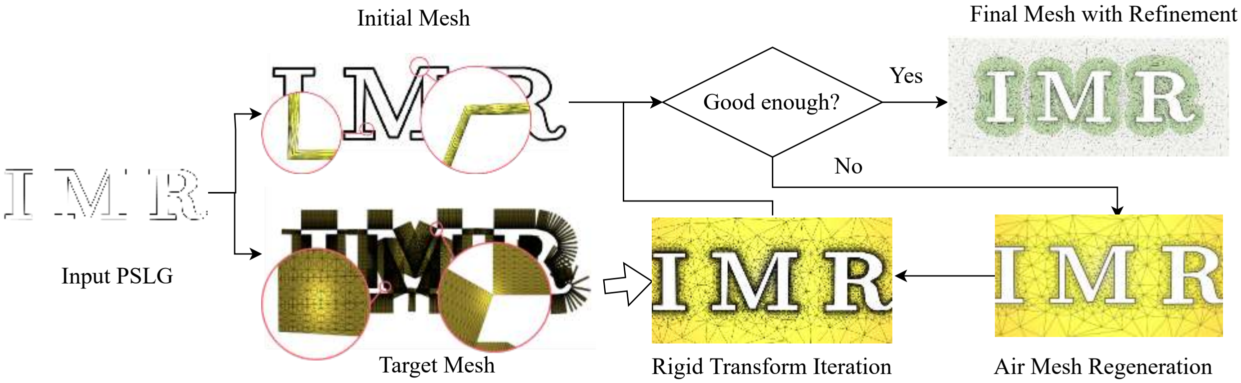


Figure 2: Overview of the proposed method (2D).

and arranged for easy visualization. All target mesh cell has no neighbor relationship since it is only used to define the "expectation", which means the deformation target.

3. **Air Mesh Regeneration** Auxiliary 2-simplicies/3-simplicies $\mathcal{M}_A = (\mathcal{V}_\mathcal{A}, \mathcal{F}_\mathcal{A})$ are used to fill the domain between the bounding box and the boundary layer mesh. The primary purpose of the application of air mesh is to prevent the fold and self-intersection of the boundary layer mesh in subsequent iterations. We only need to maintain the positivity of the volume of the air mesh cell during the iteration process. The idea of auxiliary 2-simplex/3-simplex comes from the *Air Meshes* [26], which is widely used for collision handling. After each iteration, the air mesh may need to be regenerated to improve its quality.

4. **Rigid Transform Iteration** The quadrilateral/prismatic meshes in the initial mesh and the target mesh have the same number of simplices $||\mathcal{F}|| = ||\mathcal{F}'||$. The purpose of the transformation is to minimize the rigid mapping energy between the initial and target mesh to make them "similar". The word "rigid" means the meshes are similar in size and shape.

5. **Final Mesh With Refinement** At the end of the process, the air mesh can be discarded, and an unstructured high-quality isotropic mesh can be generated around the boundary layer mesh. The quality of the air mesh is limited since it is only used to avoid intersections of the boundary layer mesh.

It is worth noting that in order to simplify the algorithm, the authors used 2-simplex/3-simplex Jacobi in 2D and 3D, respectively, to compute the energy. Therefore, every quadrilateral mesh in both the initial mesh and target mesh is decomposed into $2 \times 2 = 4$ triangles as shown in Figure 3. Compared with only decomposition into two triangles by diagonals, this decomposing scheme has its advantage: we only need to preserve the positivity of the triangle's area to ensure the convexity of the final quadrilateral mesh. In 3D, similarly, a triangular prism can be decomposed into three tetrahedrons, as shown in Figure 4. Because there are six possible schemes for decomposing a triangular prism into tetrahedrons, every triangular prism is decomposed into $6 \times 3 = 18$ tetrahedrons with overlapping regions in implementation.

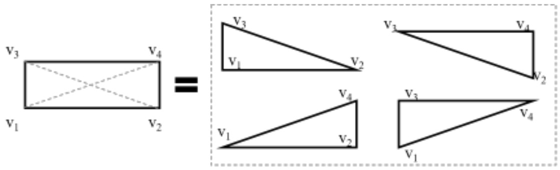


Figure 3: Decomposition of a quadrilateral mesh.

3. INITIAL MESH AND TARGET MESH GENERATION

3.1 Initial Mesh Generation

The existence of the initial mesh is the fundamental guarantee of robustness. Similar to conventional ALM, the initial mesh depends on the marching normal and distance. The critical point is that no self-intersection is allowed in the initial mesh. For the marching distance, we can prove that the mesh is free of fold and intersection as long as the marching step size is small enough. Algorithm 1 shows the procedure

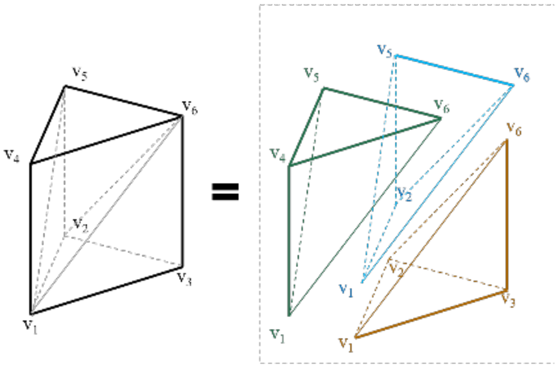


Figure 4: One of the decomposition schemes of a prismatic mesh.

of generation:

Algorithm 1 Initial Mesh Generation

```
Calculate the initial normal.
Generate only one thick layer boundary layer mesh following the fixed initial normals and initial marching step length $H_{all}$.
while there exists fold or self-intersection in the outermost loop/surface in 2D/3D do
    $H_{all} = 0.5 * H_{all}$.
    Split the one layer boundary layer mesh into initial mesh.
```

For the marching normal, in 2D, a reasonable choice for point normal is the average of neighbor front normal. In 3D, the "most normal" [29] is introduced for calculation. Sometimes one normal may not be enough for extremely complex corners, and a multiple normals configuration must be introduced to solve the problem of the existence of the initial mesh.

3.2 Target Mesh Generation

The target mesh is the combination of ideal mesh elements, which define the target of iteration. The design of the target mesh will directly determine the effect of the final mesh. The intuitive idea is that the corresponding input PSLG's segment length determines the horizontal size. The vertical size, in turn, is determined by the user input parameters, including the first layer's height, height ratio, and layer number.

This intuitive idea may lead to low-quality mesh in narrow gap areas. Figure 5 shows the target mesh of the model and its corresponding boundary layer mesh and compares the effect with and without adaptive target mesh adjustment. Because the large area target mesh and narrow gap are incompatible, the rigid transformation algorithm must balance the mesh quality and area by increasing distortion.

The target mesh in 2D is rectangular. Since the target mesh is rotationally-invariant, it has only two degrees of freedom: the horizontal size H and the vertical size V, i.e., $\mathcal{V}' = (V, H)$.

Horizontal The horizontal size is the rectangular mesh that parallels the boundary. In general, the horizontal size of segment/facet e is decided by the initial size H_e^0 and the ideal size $H_e^{k_{max}}$, where k_{max} is the maximum number of layers. H^0 is equivalent to the length of the corresponding segment in the input PSLG, while $H_e^{k_{max}}$ can be obtained by the Laplace smoothing with H_e^0 as the initial value. Finally, the horizontal size of k_{th} layer can be linearly defined as:

$$H_e^k = \frac{kH_e^{k_{max}}}{k_{max}} + \frac{(k_{max} - k)H_e^0}{k_{max}} \tag{1}$$

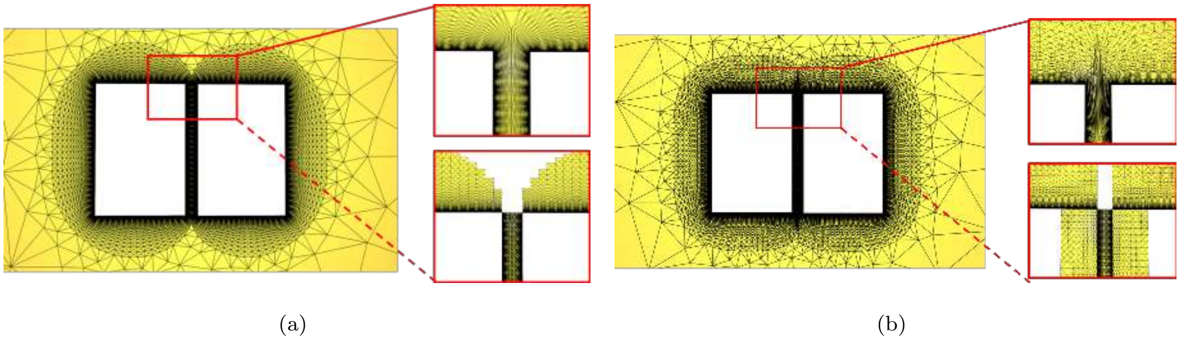
Figure 6 shows the example of target mesh after adjusting the horizontal size; a pronounced sawtooth can be observed between the adjoint edge of two rectangular.

Vertical The vertical size defines the height of the target mesh. Figure 5 shows the comparison between with and without adaptive vertical size adjustment. Figure 5(b) shows the fixed target mesh and its corresponding final mesh after infinite iteration. Some twisted elements are generated since the narrow gap constraint. Figure 5(a) shows the target and final mesh after adjustment, and twisting is alleviated. We can also see that the corresponding target mesh is compressed.

The vertical size adjustment of the target mesh is usually achieved by shrinking the step length. An overly aggressive shrinking step length strategy usually results in a slow convergence, while an overly loose strategy cannot achieve the desired goal. The author proposes a strategy of shrinking layers. For segment e in each iteration, suppose the ideal vertical height calculated by user input as $V_e^{ideal} = \sum_{k=0}^{k_{max}} \alpha\gamma^k$, where α is the height of the first layer, and γ is the growing ratio. Then, the distance between the 0 layer and k_{max} layer center of e as $V_e^{current} = \frac{1}{dim}\sum_{j=1}^{dim} ||v_{e,j}^0 - v_{e,j}^{k_{max}}||$, where $v_{e,j)}$ denoted the coordinate of vertex of simplex e. We can calculate the vertical size in the next iteration as follows:

$$V_e^k = \sqrt{V_e^{current} V_e^{ideal}} \tag{2}$$

Since equation 2 is related to the current height, after each iteration, the target mesh needs to be recalculated, including the gradient. Obviously, $V_e^k > V_e^{current}$, so the height of the target mesh is higher than the height of the existing iteration mesh. After several iterations, this value will eventually converge.



(a)

(b)

Figure 5: Illustrative example of the adjustment of the vertical size of target mesh and final mesh. The subfigure connected by the solid line in the upper right corner of each figure shows the zoom-in view of triangles, and the subfigure connected by the dotted line in the lower right corner shows the corresponding target mesh (a) the final mesh with vertical size adjustment. (b) the final mesh without vertical size adjustment.

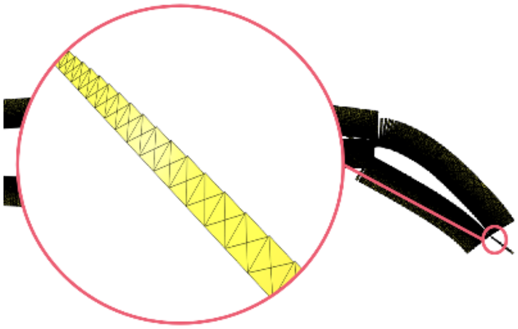


Figure 6: Illustrative example of the adjustment of the horizontal size of the target mesh.

3.3 Multiple Normals Configuration

Multiple normals configurations enhance the initial mesh generation in both 2D and 3D. The configuration adopts the notion of virtual input for PSLG. In an extremely sharp convex point, a segment of zero length is inserted at the sharp corners of the PSLG, i.e., extra coincident points are generated at the sharp corners so that the subsequent algorithm requires only slight modification.

Figure 7 shows the mesh comparison after infinite iterations between with and without multiple normal configurations. An obvious extra normal can be observed in Figure 7(a) compared with Figure 7(b). Figure 7(c) shows the target mesh with multiple normal; We can observe three extra target mesh straps marching from the degenerated segment. It is worth noting that degenerate triangles are discarded for the first layer of extra straps of the degenerate segments, and only one triangle is generated instead.

4. RIGID MAPPING

4.1 Problem Statement

Suppose the rigid mapping can be defined as $\phi : \mathcal{V}' \to \mathcal{V}$, our target is to minimize the mapping energy:

$$\begin{aligned} &\min_{\mathcal{V}} E\left(\phi\right) \\ &s.t. \mathcal{M} \text{ is self-intersection free.} \end{aligned} \tag{3}$$

Air mesh [26] is widely used to solve the global intersection problem of $\mathcal{M}$, while the $\mathcal{M}$ is local intersection free if there are no flipped or negative *oriented area* triangles in it. $\mathcal{M}_A$ share the same nodes in the outermost layer $\mathcal{M}$. Therefore, the $\mathcal{M}_A$ without fold is equivalent to the $\mathbf{M}$ without global self-intersection. Formally, suppose $A(t)$ is the oriented area of triangle (simplex) t, the problem 3 can be rewritten as:

$$\begin{aligned} &\min_{\mathcal{V}} E\left(\phi\right) \\ &s.t. \forall t \in \mathcal{M}, A\left(t\right) > 0 \\ &\quad \forall t_A \in \mathcal{M}_{\mathcal{A}}, A\left(t_A\right) > 0 \end{aligned} \tag{4}$$

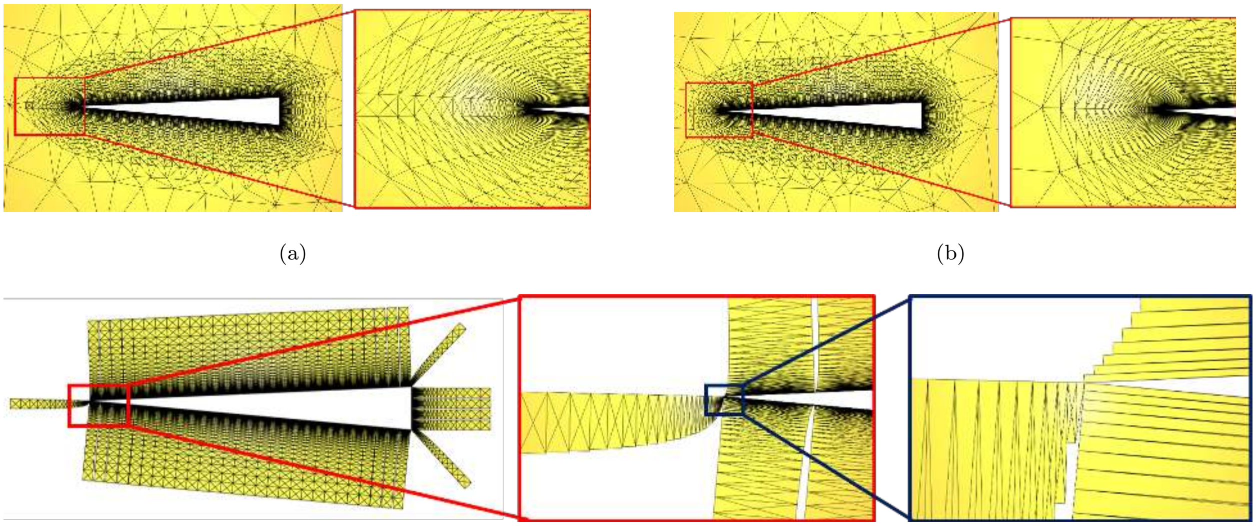
The energy function determines the shape of the final mesh. And proper energy function and precise control of step size by line search [30], e.g., energy function with zero barriers, can prevent negative simplex.

4.2 Energy Definition

4.2.1 Rigid Mapping Energy

Rigid distortion energy has been well-studied in mesh deformation and surface parameterization. Suppose the Jacobian of the map ϕ computed from each simplex $f \in \mathcal{F}$:

$$\mathbf{J}_f := \nabla \phi_f \tag{5}$$



(a)

(b)

(c)

Figure 7: The final mesh and target mesh of a sharp convex example with and without multiple normals configuration. (a) the final mesh with multiple normal generates an extra normal on the convex point. (b) the final mesh without multiple normals. (c) the target mesh with multiple normal, the mesh size of the extra segment gradually increases with height. In particular, the first layer's target mesh only has one triangle.

Where ϕ_f is the restriction of ϕ over the simplex f, which is an affine map. Then we can denote the energy as:

$$E\left(\phi\right) = \sum_{f \in \mathcal{F}} \mathcal{D}\left(\mathbf{J}_f\right) \tag{6}$$

Where $\mathcal{D}(\cdot)$ is the distortion energy of each simplex. The slight difference from the application in parameterization or mesh deformation is that we do not need weight coefficients such as mesh area, etc. The physical properties of the boundary layer mesh determine that the unweighted equation is enough.

Generally, conforming mapping contains those, whether preserving the angle or preserving the scale, and under the boundary constraint, these two conditions conflict. The most popular form of rigid mapping energy called "as-rigid-as-possible" (ARAP) mapping was proposed by Liu *et al.* [31], which is a famous mapping method that balances those two conditions, and the distortion energy of it can be defined as:

$$\mathcal{D}_{\mathcal{ARAP}}\left(\mathbf{J}_f\right) = ||\mathbf{J}_f - \mathbf{R}\left(\mathbf{J}_f\right)||_F^2 \tag{7}$$

Where $\mathbf{R}(\mathbf{J}_f)$ is the closest rotation to $\mathbf{J}_f$, and $|| \cdot ||_F$ denotes the Frobenius norm. The idea of finding the closet rotation $\mathbf{R}$ is that if we use SVD to decomposite $\mathbf{J} = \mathbf{U}\Sigma\mathbf{V}^T$, $\mathbf{R}(\mathbf{J}) = \mathbf{U}\mathbf{V}^T$ as a rotate matrix, while Σ stand for scaling. One of the advantages of this energy is that it can be performed by the local/global method [31].

4.2.2 Local/Global Iteration

The *Local/Global Iteration* method was first published by Liu *et al.* [31]. This method unprecedentedly decomposes the global deformation energy optimization into local and global linear calculations, which makes it possible to optimize the global energy by solving only linear systems. The iteration is decomposite into two steps:

1. **Local** the computation of the closest rotation to Jacobian for complex f in k iteration $\mathbf{R}_f^k := \mathbf{U}\mathbf{V}^T$. ($\mathbf{U}$ and $\mathbf{V}$ is the first and the third part of SVD decomposition of the Jacobian)

2. **Global** the computation to solve a global linear equation 7 that minimizing the distortion energy as:

   $$\arg\min_{\mathcal{V}^k} \left(\sum_{f \in \left(\mathcal{F} \cup \mathcal{F}_{\mathcal{A}}\right)} ||\mathbf{J}_f\left(\mathcal{V}^k\right) - \mathbf{R}\left(\mathbf{J}_f\left(\mathcal{V}^{k-1}\right)\right)||_F^2 \right) \tag{8}$$

   This equation is simple quadratic energy if $\mathbf{R}$ is known since we can rewrite it as the most celebrated weight recipes are the so-called *cotangent weights* [32]. Thus the extreme point of the energy can be obtained by solving a linear equation.

4.2.3 Symmetry Dirichlet Energy

According to the author's experiment, $\mathcal{D}_{\mathcal{ARAP}}$ did not apply to the boundary layer mesh. In this manuscript, we chose one of its variants: another rigid mapping energy called *Symmetry Dirichlet Energy* proposed by Jason Smith [30] is utilized to measure the distortion, which has been proved rotation invariant:

$$\mathcal{D}_{\text{SDE}}\ (\mathbf{J}_f) = ||\mathbf{J}_f||_F^2 + ||\mathbf{J}_f^{-1}||_F^2 = \sum_i^{dim}\ (\sigma_i^2 + \sigma_i^{-2}) \quad (9)$$

Where σ_i is the eigenvalue of Σ, or say the singular value of $\mathbf{J}$, *dim* means the dimension of the problem. Apparently, the equation 9 is singular when σ is taken to 0. Moreover, the geometrical meaning of $\sigma = 0$ is that the triangle has completely degenerated, and the area is 0.

In order to take advantage of the local/global method while using the symmetry Dirichlet energy, Jiang *et al.* [22] proposed the *Weighted Proxy Functions*, which extended the method to the anisotropic weights. We can rewrite the distortion measure as follows:

$$\mathcal{D}_{\text{SDE}}^{\mathbf{W}}\ (\mathbf{J}_f) = ||\mathbf{W}\ (\mathbf{J}_f - \mathbf{R}\ (\mathbf{J}_f))||_F^2 \quad (10)$$

Where $\mathbf{W}$ is the 2 x 2 proxy matrix. The proxy matrix arbitrary energy $\mathcal{D}\ (\mathbf{J})$ can be written as:

$$\mathbf{W} = \ (\frac{1}{2}\nabla_{\mathbf{J}}\mathcal{D}\ (\mathbf{J})\ (\mathbf{J} - \mathbf{R})^{-1})^{\frac{1}{2}} \quad (11)$$

Suppose $\mathbf{J} = \mathbf{U}\Sigma\mathbf{V}^T$ is the singular value decomposition of $\mathbf{J}$ and $\mathbf{I}$ is the identity matrix. Since then the energy of equation 9 is rotate invariant, the equation 12 can be rewritten as:

$$\mathbf{W} = \mathbf{U}\ (\frac{1}{2}\nabla_{\Sigma}\mathcal{D}\ (\Sigma)\ (\Sigma - \mathbf{I})^{-1})^{\frac{1}{2}}\mathbf{U}^T = \mathbf{U}\Sigma_{\mathbf{W}}\mathbf{U}^T \quad (12)$$

For the energy of equation 9, the proxy matrix is:

$$\Sigma_{\mathbf{W}} = \ (\frac{\sigma_i - \sigma_i^{-3}}{\sigma_i - 1})^{\frac{1}{2}} \quad (13)$$

Come back to equation 10, both $\mathbf{R}$ and $\mathbf{W}$ can be calculated by the current State, thus minimizing the equation 10 is also equivalent to solving a linear equation. Therefore, local/global method is also available:

1. **Local** the computation of the $\mathbf{R}$ and $\mathbf{W}$
2. **Global** the computation to solve a global linear equation.

4.2.4 Air Mesh Energy

The intersection detection of the air mesh is also handled by equation 9. No self-intersection happens by controlling the energy away from the singularities without crossing over it in each iteration. Therefore, the two constraints of Problem 4 are both held by the equation 9.

The air mesh quality is not as important as the boundary layer mesh; thus, in each iteration, the air mesh in the last round is chosen as the target mesh of the air mesh. Formally, suppose the $\mathcal{M}_A^{k-1} = (\mathcal{V}_A^{k-1}, \mathcal{F})$ as the air mesh in $k-1$'s round of iteration, the rigid mapping of air mesh can be defined as $\psi : \mathcal{V}_A^{k-1} \to \mathcal{V}_A^k$. By putting the air mesh and boundary layer mesh into the same frame of consideration, the proposed method is implemented by optimizing the following equations:

$$\begin{aligned} &\min_{\mathcal{V}} E\ (\phi) + \lambda E\ (\psi) \\ s.t. &\forall t \in \mathcal{F}, A\ (t) > 0 \\ &\forall t_A \in \mathcal{F}_{\mathcal{A}}, A\ (t_A) > 0 \end{aligned} \quad (14)$$

Because we only care about the singularity of ψ, the choice of λ is small enough to guarantee the functionality of Air Mesh with little to no effect on the final result. In our experience, $\lambda = \frac{1}{10000||\mathcal{F}||}$, where $||\mathcal{F}||$ represents the number of simplexes in $\mathcal{F}$.

The computation of $E(\psi)$ follows the same pattern as $E(\phi)$. Because they share points on the boundary, they are calculated together, and the only difference is the weights, which are decided by the flexible factor λ.

4.3 Positive Volume Gurantee

Since the solution of equation 14 may cause flip or self-intersection(The simplex f with $\det(\mathbf{J}_f) < 0$ may still have small energy according to equation 9), the simple application of the solution is not enough. In the same way as the method used in the above papers, the authors also used the *Line-Search* to avoid self-intersection mesh. The Line-Search is a widely-used technique for optimization first published as detailed in Nocedal et al [33]. Suppose the $\mathcal{V}^k$ and $\mathcal{V}^d$ is the coordinate of k_{th} and $k+1_{th}$ iteration, and if we can guarantee that the direction of optimization must lead to energy reduction, then there must exist an α such that the energy of $\mathcal{V}^{k+1} = \alpha\mathcal{V}^k + (1-\alpha)\mathcal{V}^d$ is smaller than $\mathcal{V}^k$ while ensuring the there is no flip or self-intersection.

This is a guarantee of the robustness of the whole algorithm. Negative cells are fatal to the simulation. Also, the algorithms using line search are usually sensitive to the choice of the optimization direction.

5. POST PROCESS

5.1 Retention Layer

Generally, one of the common criticisms of the boundary layer mesh is that anisotropic meshes leave too small of a gap after generation, making high-quality isotropic mesh generation a hard problem. Therefore, the top layer of the boundary layer mesh is used as the retention layer to avoid narrow gaps. Thus, boundary layers will retain a gap height of at least two preserving layers. A control parameter β is used to control the ratio between the height of the reserved layer and the default height. Figure 8 compares the retention layer and the final mesh at $\beta = 0.1$ and $\beta = 2$.

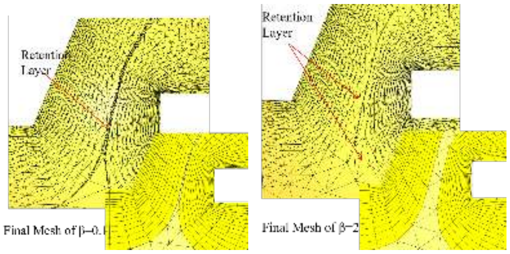


Figure 8: The retained layer schematic and final mesh at β=0.1 and 2.

5.2 Mesh Refinement

After removing the retention layer, the proposed algorithm fills the remaining domain with an isotropic mesh. Unlike the air mesh, the calculation of the size field is driven by boundaries (Figure 1), and the number of mesh cells is significantly increased, which may be more conducive to simulation calculations.

6. RESULT

6.1 IMR

The model contains three English letters, "I", "M" and "R". sharp concave corner can be found in "M", and there exists a nested ring in "R". There are both straight and curved turns in this example, as well as sharp concave corner. There is also a narrow gap at the bottom of the "M" letter, making it difficult to generate high-quality full-layer meshes. Figure 9 shows the mesh with air mesh under the different number of iterations. It can be seen that the mesh gradually expands from the initial mesh, and after about 100 iterations, the mesh gradually expands to the ideal height. There are also no extremely twisted elements in the narrow gap.

The model contains three English letters, "I", "M" and "R". sharp concave corner can be found in "M", and there exists a nested ring in "R." There are both straight and curved turns in this example, as well as sharp concave corner. There is also a narrow gap at the bottom of the "M" letter, making it difficult to generate high-quality full-layer meshes. Figure 9 shows the mesh with air mesh under the different number of iterations. It can be seen that the mesh gradually expands from the initial mesh, and after about 100 iterations, the mesh gradually expands to the ideal height. There are also no extreme, twisted elements in the narrow gap.

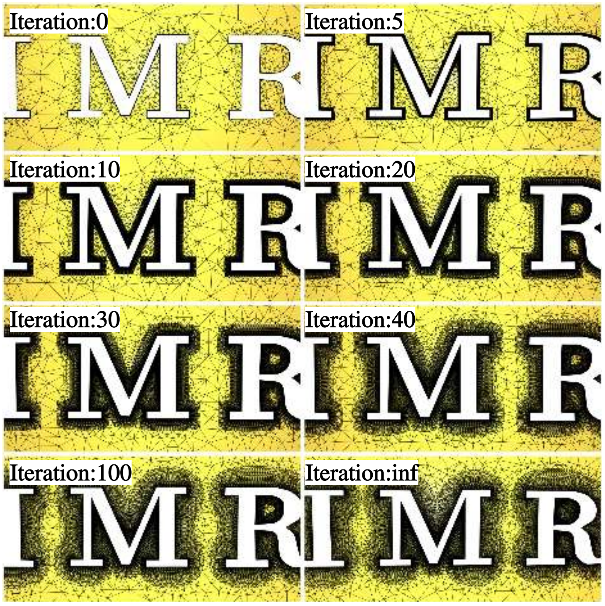


Figure 9: The initial mesh and the air mesh of the IMR English letter model by a different number of iterations.

6.2 30P-30N airfoil

To further verify the method, a complex configuration of a 2D three-element airfoil, the 30P-30N airfoil, is tested. Figure 11 shows the initial and target mesh under the different number of iterations. In this example, after 300 iterations, the mesh quality at the narrow gap increases as the height of the target mesh decreases. Figure 10 shows the final layered mesh, the details of the gaps are shown in the two subfigures at the bottom, and the algorithm in this manuscript can handle the narrow gap at the cross-region of different assemblies.

6.3 U-shape

An academic example named U-shape is introduced to demonstrate the preliminary results obtained by the program in 3D. The U-shape model is a small box obtained by the Boolean subtraction of two cubes of

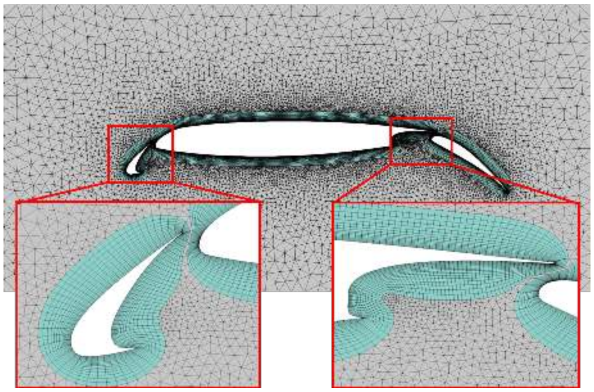


Figure 10: The final mesh of 30P-30N airfoil.

different sizes. As shown in Figure 12, the input surface mesh contains 3,102 points and 6,200 triangular meshes, and 45 layers of prismatic mesh generate initially. Figure 12 shows the final prismatic mesh after the different number of iterations. The result indicates that at least 100 iterations are required for the program to obtain a high-quality mesh.

To demonstrate the effectiveness of the proposed algorithm, the author compares the final mesh with the commercial software Pointwise, a piece of prevalent commercial software for meshing tasks with the same surface input. Figure 13 shows the cut-view comparison between the two meshes, and we can see that Pointwise cannot generate a complete mesh near the concave corner. In addition, *Equiangule Skewness*[2] is utilized to measure the mesh quality. Figure 14 shows the quality distribution of the two meshes. Since lower-quality meshes are more harmful to simulation, a logarithmically vertical axis is presented in the comparison. It can be observed that the algorithm proposed in this manuscript is ahead of Pointwise in this indicator. Moreover, the worst quality usually plays a decisive role in the simulation convergence speed and accuracy. The author compares the maximum prism equiangle skewness of the two meshes, and the results of the proposed algorithm (0.9914) outperform Pointwise (0.9504).

6.4 DLR F6(One Layer)

The method proposed in this manuscript is very time-consuming in 3D. Due to the limitation of running time, we only generate one layer thick boundary layer mesh in the F6 model. This is a challenging task for highly curved surface [34]. Figure 15 shows the comparison of the input surface mesh (the green part) and the outermost surface mesh (the white part) of the final prismatic mesh. Figure 16 shows the detailed prismatic mesh around the connection point of the aircraft hanger from different views. It can be seen that the prismatic mesh is full around complex corner points. The generation of this example takes about 15.0 hours, and the mapping energy is reduced from 1.5×10^{16} to 6.5×10^{8}. A single-layer mesh is simulation meaningless, but it may open a door for large-scale full-layered prismatic mesh generation.

7. CONCLUSION AND LIMITATION

This article presents a novel robust method for full-layer boundary layer mesh generation. By defining the target mesh and initial mesh Symmetry Dirichlet mapping Energy, we can gradually expand any thin initial mesh to the ideal through iterations. In addition, the proposed algorithm will dynamically adjust the size of the target mesh for better boundary layer mesh quality. We achieve good results in 2D and some preliminary results in 3D, along with multiple normal configurations.

However, the current version of the proposed method in 3D has a huge time bottleneck. This is because $\mathcal{V}^k$ of equation 8 needs to be obtained by solving a linear equation. This linear equation system has a sparse matrix with a row of (number of prisms × 18) and a column of (number of mesh points). Considering the boundary layer meshes in real industry, the quantity of mesh often exceeds one million. Whether the iterative or direct solution is used, it will be very time-consuming to solve this linear equation system. In addition, its convergence speed is also relatively slow, and it takes more than 500 iterations to obtain good results. Though the current version has a time bottleneck, the proposed algorithm may have engineering potential due to its robustness if we can overcome the time performance issue in the future.

References

[1] Middlecoff J., Thomas P. "Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations." *AIAA Journal*, vol. 18, Feb. 1979

[2] Steger J., Sorenson R. "Use of Hyperbolic Partial Differential Equations to Generate Body Fitted Coordinates." NASA. Langley Research Center Numerical Grid Generation Tech, Feb. 1980

[3] Thompson J. "A General 3D Elliptic Grid Generation System on a Composite Block Structure." *Computer Methods in Applied Mechanics and Engineering*, vol. 64, 377–411, Oct. 1987

[2] https://www.pointwise.com/doc/user-manual/examine/functions/equiangle-skewness.html

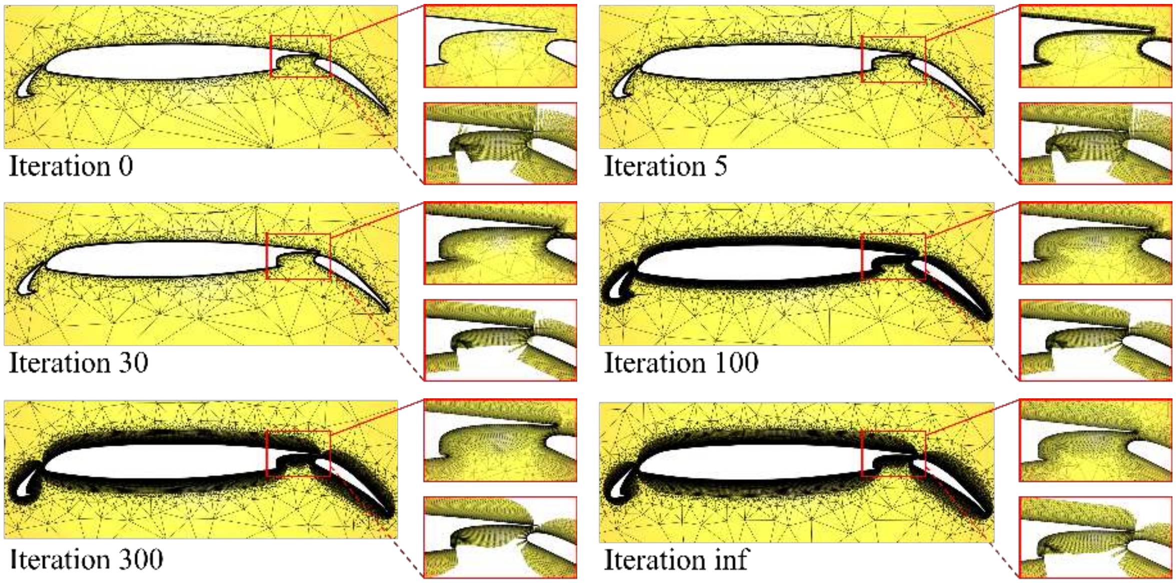


Figure 11: The initial mesh and the air mesh of the 30P-30N airfoil model by different iterations. The subfigure connected by the solid line in the upper right corner of each figure shows the mesh, and the subfigure connected by the dotted line in the lower right corner shows the corresponding target mesh.

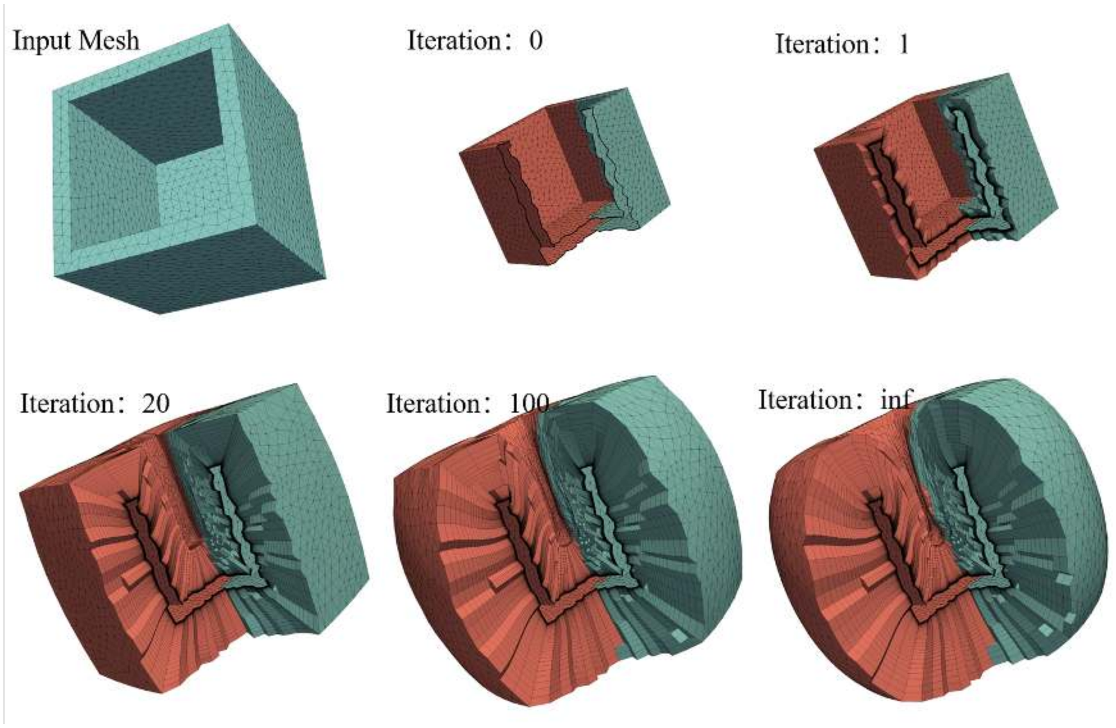


Figure 12: The surface mesh and the layered boundary layer mesh of U-shape model by the different number of iterations.

[4] Sadrehaghighi I. *Mesh Generation in CFD*. No. Patch 1.86.7 in CFD Open Series. ANNAPOLIS, Jan. 2020

[5] Baker T.J. “Mesh Generation: Art or Science?” *Progress in Aerospace Sciences*, vol. 41, no. 1, 29–63, 2005

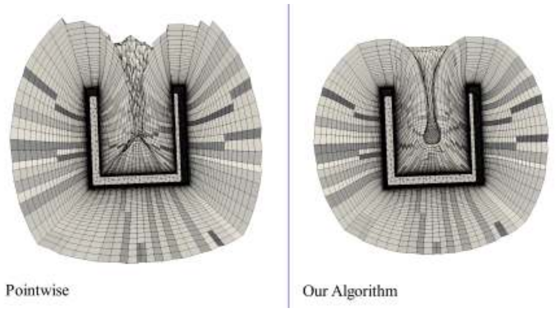


Figure 13: Meshes generated by the proposed algorithm and Pointwise.

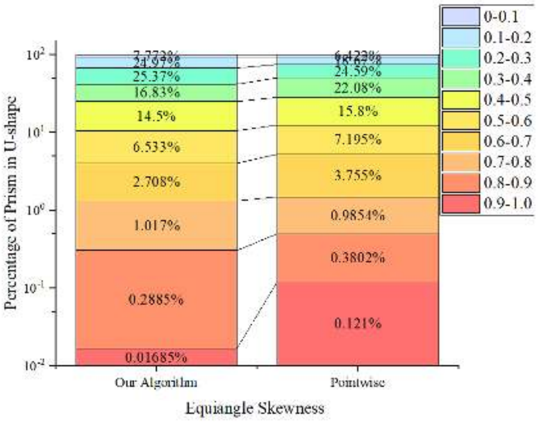


Figure 14: comparison of mesh quality of meshes generated by the proposed algorithm and Pointwise.

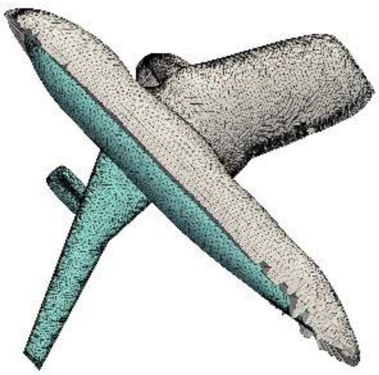


Figure 15: comparison of input surface mesh and single-layer prismatic mesh.

[6] Garanzha V., Kudryavtseva L., Belokrys-Fedotov A. “Single and Multiple Springback Technique for Construction and Control of Thick Prismatic Mesh Layers.” *Russian Journal of Numerical Analysis and Mathematical Modelling*, vol. 36, 1–15, 2021

[7] Zheng Y., Xiao Z., Chen J., Zhang J. “Novel Methodology for Viscous-Layer Meshing by the Boundary Element Method.” *AIAA Journal*, vol. 56, no. 1, 209–221, 2018

[8] Zhu Y., Wang S., Zheng X., Lei N., Luo Z., Chen B. “Prismatic Mesh Generation Based on Anisotropic Volume Harmonic Field.” *Advances in Aerodynamics*, vol. 3, no. 1, 2021

[9] Wang Y., Guibault F., Camarero R. “Eikonal Equation-based Front Propagation for Arbitrary Complex Configurations.” *International Journal for Numerical Methods in Engineering*, vol. 73, 226–247, Jan. 2008

[10] Wang Y. “Eikonal Equation Based Front Propagation Technique and Its Applications.” *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*, Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, Jan. 2009

[11] Xia H., Tucker P.G., Dawes W.N. “Level Sets for CFD in Aerospace Engineering.” *Progress in Aerospace Sciences*, vol. 46, no. 7, 274–283, 2010

[12] Wang Y., Murgie S. “Hybrid Mesh Generation for Viscous Flow Simulation.” P.P. Pébay, editor, *Proceedings of the 15th International Meshing Roundtable*, pp. 109–126. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006

[13] Dyedov V., Einstein D.R., Jiao X., Kuprat A.P., Carson J.P., Del Pin F. “Variational Generation of Prismatic Boundary-Layer Meshes for Biomedical Computing.” *International Journal for Numerical Methods in Engineering*, vol. 79, no. 8, 907–945, 2009

[14] Roget B., Sitaraman J., Lakshminarayan V., Wissink A. “Prismatic Mesh Generation Using Minimum Distance Fields.” *Computers & Fluids*, vol. 200, 104429, 2020

[15] Wang F., Mare L. “Hybrid Meshing Using Constrained Delaunay Triangulation for Viscous Flow Simulations.” *International Journal for Numerical Methods in Engineering*, vol. 108, no. 13, 1667–1685, 2016

[16] Eller D., Tomac M. “Implementation and Evaluation of Automated Tetrahedral–Prismatic Mesh Generation Software.” *Computer-Aided Design*, vol. 72, 118–129, 2016

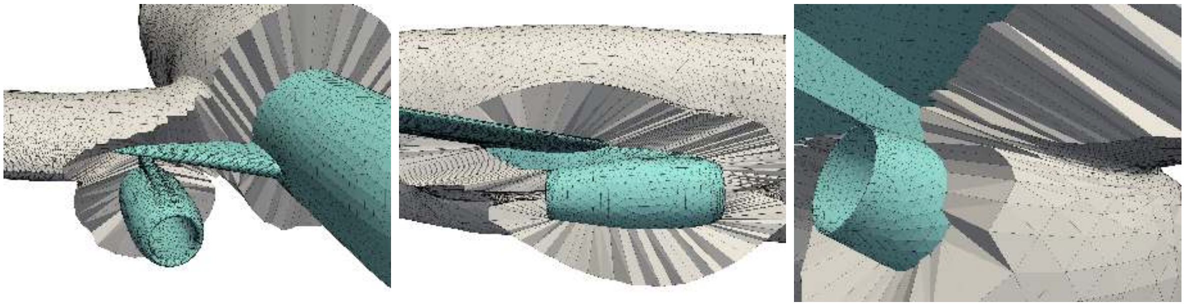


Figure 16: The details of the single-layer prismatic mesh of f6.

[17] Ye H., Liu Y., Chen B., Liu Z., Zheng J., Pang Y., Chen J. “Hybrid Grid Generation for Viscous Flow Simulations in Complex Geometries.” *Advances in Aerodynamics*, vol. 2, no. 1, 2020

[18] Loseille A., Löhner R. “Robust Boundary Layer Mesh Generation.” X. Jiao, J.C. Weill, editors, *Proceedings of the 21st International Meshing Roundtable*, pp. 493–511. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013

[19] Weinstein A. “Theoretical Kinematics (O. Bottema and B. Roth).” *SIAM Review*, vol. 22, 519–520, Oct. 1980

[20] Botsch M., Sorkine O. “On Linear Variational Surface Deformation Methods.” *IEEE transactions on visualization and computer graphics*, vol. 14, 213–30, Jan. 2008

[21] Jiang Z., Schaefer S., Panozzo D. “Simplicial Complex Augmentation Framework for Bijective Maps.” *ACM Transactions on Graphics*, vol. 36, 1–9, Nov. 2017

[22] Rabinovich M., Poranne R., Panozzo D., Sorkine-Hornung O. “Scalable Locally Injective Mappings.” *ACM Transactions on Graphics*, vol. 36, 1, Apr. 2017

[23] Sorkine O., Alexa M. “As-Rigid-As-Possible Surface Modeling.” *Symposium on Geometry processing*, vol. 4, 109–116, Jan. 2007

[24] Alexa M., Cohen-Or D., Levin D. “As-Rigid-As-Possible Shape Interpolation.” *Proceedings of SIGGRAPH 2000*, Feb. 2000

[25] Xu D., Zhang H., Wang Q., Bao H. “Poisson Shape Interpolation.” *Graphical Models*, vol. 68, 268–281, May 2006

[26] Müller M., Chentanez N., Kim T., Macklin M. “Air Meshes for Robust Collision Handling.” *ACM Transactions on Graphics*, vol. 34, Aug. 2015

[27] Zhang E., Mischaikow K., Turk G. “Feature-Based Surface Parameterization and Texture Mapping.” *ACM Transactions on Graphics*, vol. 24, Jan. 2005

[28] Barill G., Dickson N., Schmidt R., Levin D., Jacobson A. “Fast Winding Numbers for Soups and Clouds.” *ACM Transactions on Graphics*, vol. 37, 1–12, Jul. 2018

[29] Aubry R., Löhner R. “On the ‘Most Normal’ Normal.” *Communications in Numerical Methods in Engineering*, vol. 24, no. 12, 1641–1652, 2008

[30] Smith J., Schaefer S. “Bijective Parameterization with Free Boundaries.” *ACM Transactions on Graphics*, vol. 34, 70:1–70:9, Jul. 2015

[31] Gotsman C., Liu L., Zhang L., Xu Y., Gortler S. “A Local/Global Approach to Mesh Parameterization.” *Computer Graphics Forum*, vol. 27, Jul. 2008

[32] Pinkall U., Polthier K. “Computing Discrete Minimal Surfaces and Their Conjugates.” *Experimental Mathematics*, vol. 2, Nov. 1996

[33] Nocedal J., Wright S.J. *Numerical Optimization.* Springer, New York, 1999

[34] Garanzha V., Kaporin I., Kudryavtseva L., Protais F., Ray N., Sokolov D. “Foldover-Free Maps in 50 Lines of Code.” *ACM Transactions on Graphics*, vol. 40, 1–16, Aug. 2021